Key Questions

Questions	Owner	Status	Writers
MPI Limitations			
1. Is there a chance that on all common future architectures (e.g. many-core nodes), standard MPI will continue to work reasonably well?	Plimpton		
2. What can we do to improve MPI performance on multicore?	Heroux		
MPI replacements/complements and transition			
In the first years of distributed memory HPC there were several different APIs that applications had to support to provide parallelism across multiple HPC platforms. The community has, for the most part, converged on MPI Version 1 as the API for distributed memory parallelism. 3. What are the leading APIs for HPC shared memory parallelism nested within distributed memory parallelism? 4. What are the availability, complexity, and performance trade-offs for the associated programming models?	Edwards		
5. Are shared memory languages of any use at all in highly scalable computing? What is the limit of their scaling and efficiency? 6. Can we find a manageable alternative to cache coherency for PGAS? 7. What can be done to improve scalability of coherency and cost of synchronization primitives?	Camp		
8. Some people are advocating highly threaded approaches and a re-look at functional languages for highly scaled problems? Is there any “there” there?	Camp		
9. If MPI only is not enough, what are the minimum changes I will need to make to my application (particle simulations) to get reasonable performance using whatever "replaces" MPI on those platforms?	Plimpton		
10. Can a portable means of directly managing access to memory bandwidth and floating point cores be provided so as to allow the average high level language code developer the ability to extract the maximum possible performance from the hardware? Constraint: Success or failure should come with immediate feedback to the developer in some way. Comments: The reason MPI is such a success is that it requires the programmer to deal directly with the parallelism and one can	Robinson		

Questions	Owner	Status	Writers
easily build tests to determine success or failure. Can this be done for on-node memory bandwidth and core usage? Remember the old Cray compilers? They would tell you whether you would run fast or not. This is an example of what I am talking about. Can a programming/memory layout system be developed that essentially ensures good performance? Strongly typed languages made a huge gain in software productivity. Can a strongly performance oriented programming model be developed?			
11. What specific, concrete productivity improvements can/will new programming models provide?	White		
12. Is it possible to gain 2x or more speedup of parallel MD by going to a non-MPI approach? 13. What are the major downsides to a non-MPI approach? Portability? Programming difficulty? Other? 14. Under what circumstances would a multi-level parallelism make sense, where MPI is used internode and something else is used intranode?	Crozier		
15. Are there extensions that can be made to MPI so that MPI is more amenable to writing scalable applications and to building next-generation libraries and languages?	Dietz		
16. What alternatives to MPI (existing or proposed) could provide better support for applications with very irregular communication patterns and highly variable work loads (discrete event simulations, dataflow (process network) systems, etc.)? In particular: * What alternatives exist for stateless and/or connectionless communication mechanisms (portals or higher level)? * How about "developer-friendly" interfaces to RMDA that go beyond what is provided by MPI-2 (and are less painful than Infiniband verbs or DAPL)?	Adalsteinsson		
Many feel that the impact on applications due to many-core and hybrid architectures will be as large as (if not larger than) the shift from vector to massively-parallel 17. Will existing applications be able to evolve to adequately take advantage of these architectures, or will the changes required be too structural, requiring entirely new code efforts? 18. One major difference in this transition is that while most applications could retain essentially a SPMD model for the vector->parallel shift, this shift may introduce more asynchronous activity than most apps have previously dealt	Turner		

Questions	Owner	Status	Writers
with - what impact will this have on existing apps, particularly large multi-physics apps? Another major difference in this transition is the variety and complexity of architectural options that are appearing - it is far greater than for vector->parallel, and some of the paths have quite different requirements on data structures and algorithm design To make matters worse, vendors are thus far (understandably) focused on software tools for their hardware (e.g. Intel's Thread Building-Blocks and Ct, NVIDIA's Cuda, IBM's ALF and DaCS, etc.) 19. In the face of these realities, how can developers best develop apps to be agnostic to the underlying hardware and still achieve acceptable performance across platforms? While data locality and movement have been important for some time, many apps have been able to ignore the issue - that will become increasingly difficult to do, and it is difficult to imagine software tools being able to perform (or even help with) many of the application and algorithm changes required for adequate performance on future platforms 20. Is this pessimism unfounded? If so, what software tools are available or under development that might help? 21. Are we at (or nearing) the end of the "MPI-everywhere" programming model? If so, what will replace it? • MPI + threads (explicit, OpenMP, TBB, ALF, etc.)? • DARPA/HPCS language (Chapel, Fortress, X10)? • Partitioned Global Address Space (PGAS) approaches (UPC, CoArray Fortran)?			
22. As a library developer, what is a recommended portable way for us to write parallel code that should run well on "any" future parallel system. (Something higher level than MPI desired.) 23. HPC is small compared to the commercial software market. What are commercial leaders like Microsoft and Google doing to prepare for an era of multicore/manycore parallelism, and how will this affect the scientific HPC world?	Boman		
24. In multicore processors, how do shared caches affect how we develop high-performance applications? What do we need to do to accommodate shared caches? Or, what do we need to do to exploit shared caches?	Chow		
25. Large portions of applications run well in MPI-only mode	Heroux		

Questions	Owner	Status	Writers
on multicore processors. Can we mix MPI-only with MPI+manycore in the same program?			
Algorithm challenges and directions			
26. Modeling and Simulation is transforming from single-run simulation to pervasive use of design optimization and uncertainty quantification. What algorithmic and architectural innovations are required to achieve this?	Collis		
27. Today's computers are less reliable than earlier generations for the simple reason that they are much more complex with many more parts—both HW and SW. When we go to a billion threads in 2018, it will be phenomenally harder for HW to be reliable. 28. Google has used the kinds of dynamic master-slave parallelization schemes that Jim Tomkins, John VanDyke, Mark Seager and Bob Benner pioneered nearly 20 years ago on the nCUBE, in this case to parallelize a whole slew of map-reduce algorithms for large data search and sort problems. Their approach is highly fault tolerant. Now, I would claim that they have an easier problem. However, in scientific computing, I have yet to see a real application come forward that is both resilient and efficient. It seems to me that I know how to do this for some cases. What is holding back the development of completely fault tolerant parallel algorithms and applications? Can we make it a priority to develop such algorithms and codes just as we made it a priority to develop the MPP SW suites that we are living on today? If so what is the prognosis for success? 29. Will hierarchical problem decomposition (I call it fractal or self-similar computing) get around the billion thread programming problem (nobody is smart enough to develop billion thread codes that do anything significant)?	Camp		
30. How much can the application be changed to support a new architectural feature? How much performance improvement has to accompany that? For each of the architectural enhancements below, score them on their usefulness to you, your likelihood of using them if they were provided, and your willingness to spend more power/money/etc to get them: • drastically more threads per core to tolerate memory and	Underwood		

Questions	Owner	Status	Writers
network latencies • Cray style vector instructions • “more” memory bandwidth (how much more) • a “small” high bandwidth, low latency user programmable memory “near” the processor (in addition to the “normal” memory) • fine grained synchronization within a processor socket • higher network bandwidth • higher “message rate” (better performance for small messages) • lower network latency			
31. How do we reduce or hide latency for global reductions and many-to-many communication patterns?	White		
32. What do users want from libraries that they don’t have now? – Functionality • Operations • Types/precisions/data layouts/ • New algorithms / helping users with algorithm choice – Automatic choice vs consulting vs education –Ease of use • Portability • Interoperability – Mixing MPI / Shared memory • Reproducibility • Maintainability – Spend 50% time helping users. Automation will not help. • Installability • Languages (native vs wrappers) • Fault tolerance • Memory models (Distributed, shared, PGAS) – Scalability • Target platforms (petascale, multicore, clusters, …) • Fraction of peak • Memory hierarchies / Out-of-core • Hierarchical machines -> hierarchical algs & SW – Standards to simplify… • Interfaces • Mixed shared / distributed memory	Dongarra		
33. What options do we have for improved fault tolerance (relocatable processes, checkpointing into neighbor memory, etc), and how invasive do these need to be on application codes?	Adalsteinsson		

Questions	Owner	Status	Writers
Other Topics			
34. Role of Automatic code generation and tuning? – When is it worth starting over to write a library generator rather than a library? • Dealing with hiearchical machines – Maintainability • Invest now for longer term reduction in costs/effort – Adapting to new architectures – How much are users willing to accommodate runtime tuning in their applications? 35. What is the role of vendors / SW companies – What do they build, what do we build? – What do they support us to build? – Multicore as opportunity to fund building some kernels – Open source and/or proprietary • Licensing (LGPL vs mBSD) Tools for future – Scalability testbed (eg RAMP) – Reproducibility	Dongarra		
36. Can we define what we mean by "performance" and what we mean by "productivity" and which is more important?	Norton		
37. Considering the operating systems currently used on capability platforms, what new or additional functionality is needed to meet the needs of application developers? Are these appropriate for systems containing millions of processor cores? 38. How should the intra-node mapping of computation to cores be handled? By the application developer? By the operating system? Libraries? Hardware? All of the above?	Pedretti		
39. What can we expect from compilers for supporting multicore?	Numrich/Heroux		